Remarks/Arguments

Office Action Summary

Status.

1. This *RESPONSE A* is in answer to the Office communication mailed 06/30/2005.
2. The Office communication is non-final.
3. NA

Disposition of Claims.

4. Claims 1 - 28 are pending in the application.
5. No Claims have been allowed.
6. The rejected Claims 1 - 28 have been amended. The claims as presently presented in the application (ORIGINAL, CANCELLED, AMENDED per A) appear as follows under the heading CLEAN CLAIMS AFTER *RESPONSE A*. The claims as presently presented in the application (ORIGINAL, CANCELLED, AMENDED per A) showing changes using a word-processing compare function appear as follows under the heading INTERLINED CLAIMS AFTER *RESPONSE A*.
7. NA
8. NA

Application Papers.

9. NA
10. The drawings are objected to and a Replacement sheet has been provided.
11. The declaration has been objected to by the Examiner. The objection is traversed and in the alternative, new declaration has been provided.

Priority under 35 U.S.C. § 119.

12. NA

DETAILED ACTION

1 Claims 1-28 as amended by this *RESPONSE A* are presented for reconsideration and examination .

Oath/Declaration

2 The Examiner has held that the declaration is defective.

2.1 The Examiner's holding that the declaration is defective is believed in error. A declaration was previously filed in compliance with MPEP §§ 602.01 AND 602.02 together with a TRANSMITTAL FORM. A Post Card acknowledgment of receipt by the US Patent and Trademark Office was received. Perhaps the Examiner has made an error since, in addition to a new declaration, the Examiner also requested a copy of the "Notice of Allowance" (PTOL 85), but no "Notice of Allowance" has been issued in this application.

3 A newly signed Declaration is submitted herewith. A copy of the "Notice of Allowance" (PTOL 85) requested by the Examiner cannot be provided since no "Notice of Allowance" has been issued in this application.

Drawings

4 The drawings have been objected to.

4.1 Regarding FIG 1, Paragraph [0022] of the specification has been amended to conform "executable code 10" as it appears in FIG 1 of the drawings and throughout the specification.

4.2 Regarding FIG 2, "Table 23" and "Translated Code Cache 22" have been renumbered as --Table 22-- and --Translated Code Cache 23--. Similarly, Paragraph [0030] of the specification has been corrected. The submitted drawing sheet is labeled "Replacement Sheet" .

Claim Rejections - 35 USC § 112, 2nd

5 The Examiner rejected Claims1, 11, 15 and 25 under 35 U.S.C.,112, second paragraph, as being incomplete for omitting essential steps, such omission amounting to a gap between the steps. See MPEP § 2172.01.

5.1 In making the rejection the Examiner states:

> *Regarding Claim 1*
> *The omitted steps are: step of emulating execution of the legacy instruction and steps relating to the use of modified "store instruction" upon identification. As stated in the preamble the method steps should lead to the conclusion, as disclosed in the preamble. Method steps disclosed are merely translating and identifying legacy instruction as modified store instruction. Further, relationship between the step of translating and identifying the store instruction does not clearly satisfying the preamble.*
> *Claims 11, 15 and 25 are rejected for the same reasons as claim 1 rejection above.*

5.2 Claim 1, 11, 15 and 25 have been amended to include appropriate execution based upon the alternatives of each "if "condition and hence the rejection is believed overcome.

6 The Examiner rejected Claims1, 11, 15 and 25 under 35 U.S.C.,112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

6.1 In making the rejection the Examiner states:

> *Regarding Claim 1*
> *The preamble in claim 1 and the subject matter that applicant claims to be his invention are not clearly tied together. Applicant discloses emulating execution of the legacy instruction in the preamble. As best understood by examiner, this merely identifies the modified store instruction, but provides no further explanation as to how emulating the execution of this modified store instruction is any different than emulating the execution of any other legacy instruction.*
> *Further, claim 1 has an incomplete decision tree. The claims discloses 3 level decision tree:*
> *if the instruction is store instruction [Else omitted]*
> *-if the afore mentioned instruction has instruction data [Else by pass check]*
> *--if the aforementioned instruction data is modified [Else omitted]*

Hence it is unclear what method steps are need in case "else" conditions are encountered. Further, it is unclear what happens when aforementioned modified instruction data is identified (last if scenario).
Further, claim 1 is rejected, as examiner is not clear what exactly the phrase "translation information" encompasses. The specification discloses translation information only in the abstract and the claims, defining it with only providing "translation details". No further explanation on the translation information is provided. i.e. structure of the translation store (which is relied upon to determine if the instruction was translated), flags etc.
Further, claim 1 is rejected, as examiner is not clear what exactly the phrase "instruction data" encompasses. Does instruction data mean the operands of the store instruction or the complete instruction itself. The specification discloses instruction data in abstract, specification paragraphs [0010] [0031] & claims, but does not define it at any other place.
All of these structural inconsistences make the claim language vague and indefinite.
Claims 11, 15 and 25 are rejected for the same reasons as claim 1 rejection above.

6.2 Claim 1, 11, 15 and 25 have been amended to more clearly identify how execution differs for store instructions and other instructions as a function of satisfying the different "if" conditions. Upon reconsideration of the amended claims, it is believed that the Examiner will now find in the claims how the execution of the "store instruction" is different than emulating the execution of any other legacy instructions.

6.3 Claim 1, 11, 15 and 25 have been amended to include appropriate execution based upon the alternatives of each "if "condition and hence the rejection is believed overcome.

6.4 The term "translation information " has been deleted from the claims and hence is no longer believed grounds for rejection.

6.5 The term "instruction data" is used in the specification in paragraphs [0010] and [0031] and is also defined by examples in other paragraphs. The meaning of "instruction data" is data that if stored (modified) will potentially cause an instruction modification (sometimes referred to as a "code modification"). It is common to refer to information in computers as being either "*instruction information*" or "*data information*". When referring to instruction processing as in the present invention,

the term "instruction data" is often used to refer to data that relates to "*instruction information*" and does not relate to "*data information*". This distinction is made clear, for example, in paragraph [0038] and other places in the specification and claims. Accordingly, referring to paragraph [0038] in particular, it is believed clear that references to modification of "instruction data" are referring to modifications of "*instruction information*" and hence referring to modifications that will potentially cause an instruction modification or a "code modification". It is believed that the terminology employed will be readily understood by those skilled in the art of computers when interpreted in light of the specification.

Claim Interpretation

7 The Examiner has interpreted the term "Indexing table" in Claim 1 as being similar to the cache translation look-aside buffer (TLB), that indicates if the translation is present in the host code block (cache equivalent) on the cache.

7.1 The Examiner's interpretation is not believed accurate nor appropriate in the present application. The TLB is a buffer that stores full data whereas the index in the present application and claims is a table that stores only a a table that stores only a count or other comparably abbreviated summary indication of potential validity. This difference between a buffer and a table is significant in that the present invention saves a significant amount of storage and a significant amount of corresponding access time.

Claim Rejections -35 USC § 101

8 The Examiner rejected Claims 1-2, 4-10, 11-12,15-16, 18-24 & 25-26 under 35 U.S.C. §101 because the claimed invention is allegedly directed to non-statutory subject matter.

8.1 The citations of the Examiner are noted together with the suggested amends to overcome the rejection.

8.2 The independent claims have been amended to recited a "computer implemented method" and "executing said translated instructions to emulate said legacy instructions" and therefore the rejection under 35 U.S.C. §101 is believed overcome.

Claim Rejections -35 USC § 102

9 The Examiner rejected Claims 1-3,7-9,15-17 & 21-23 under 35 U.S.C. 102(e) as being anticipated by U.S. Patent No. 6,516,295 issued to George A. Mann et al (Mann '295 hereafter).

9.1 In making the rejection, the Examiner states:

Regarding Claim 1

Mann '295 teaches a method of emulating execution of a legacy instruction (Mann '295: Col.2 Lines 44-51). Mann '295 teaches instructions having instruction address (Mann '295: Col.5 Lines 28-29).

Further, Mann '295 teaches accessing blocks of legacy instruction (Mann '295: Col.6 Lines 11-12). Mann '295 teaches blocks having block addresses (Mann '295: Col.6 Lines 17-19).

Further, Mann '295 teaches storing into translation store as the host code block (Mann '295: Fig. 3 Element 88; Col.5 Lines 58-63; Col.6 Lines 11-28) for each legacy instruction.

Further, Mann '295 teaches storing translation indication at block numbers determined by block addresses (Mann '295: Fig. 3 Element 81). Mann '295 teaches indexing table as block entry table for indicating that the block has been translated (Mann '295: Col.6 Lines 62-66).

Further, Mann '295 teaches each particular legacy instruction of the translated block having a particular block number (Mann '295: Fig. 3 Element 72F-L, 80-81).

Further, Mann '295 teaches translating a particular legacy instruction into one or more translated instructions for emulating the particular legacy instruction (Mann '295: Col.6 Lines 53-55).

Further, Mann '295 teaches checking store instruction associated to the block entry table, if instruction data is stored for a particular block (Mann '295: Col.9 Lines 5-10).

Mann '295 also teaches checking if the instruction data has been modified (Mann '295: Col.9 Lines 9 -20). Mann '295 teaches bypassing the checking if there is no store instruction data (Mann '295: Fig. 5 Element 134/136 & 148).

Regarding Claim 2

Mann '295 teaches the step of storing translation indications for only a subset of all translated blocks (Mann '295: Col.5 Lines 54-63).

Regarding Claim 3

Mann '295 storing the translated executable host code on the volatile cache memory like SRAM (Mann '295: Col.3 Lines 36-38, 55-61; Col.4 Lines 16-18).

Regarding Claim 7

Mann '295 teaches that legacy instructions are object code instructions compiled/assembled for the legacy architecture (Mann '295: Col.2 Lines 44-51).

Regarding Claim 8

Mann '295 teaches legacy instruction includes store instructions for modifying instruction code (Mann '295: Col.9 Lines 5-20, 38-47; Col.7 Lines 14-37).

Regarding Claim 9

Mann '295 teaches translation indication includes a state field as tag field, for each block number, indicating the block has been modified (Mann '295: Col.5 Table 1; Col.6 Lines 62-67).

Regarding Claim 15

Mann '295 teaches an apparatus and a method for emulating self-modifying code. The system claim 15 is directed towards the same limitations as the method claim 1 and is rejected for the same reason as claim 1.

Regarding Claim 16

The system claim 16 is directed towards the same limitations as the method claim 2 and is rejected for the same reason as claim 2.

Regarding Claim 17

The system claim 17 is directed towards the same limitations as the method claim 3 and is rejected for the same reason as claim 3.

Regarding Claim 21

The system claim 21 is directed towards the same limitations as the method claim 7 and is rejected for the same reason as claim 7.

Regarding Claim 22

The system claim 22 is directed towards the same limitations as the method claim 8 and is rejected for the same reason as claim 8.

Regarding Claim 23

The system claim 23 is directed towards the same limitations as the method claim 9 and is rejected for the same reason as claim 9.

9.2 The independent Claims 1, 11 and 15 have been amended to include limitations like those of non-rejected Claim 4 and hence the arguments of the Examiner, as quoted in Section 9.1 above, are now believed moot and therefore the arguments are not traversed and are not accepted or rejected.

Claim Rejections -35 USC § 103

10 The Examiner rejected Claims 4-5,10-13,18-19 & 24-27 under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No.6516295 issued to George A. Mann et al (Mann '295 hereafter) in view of ACM Article "Cache Memories" by Alan Jay Smith (Smith '1982 hereafter).

10.1 In making the rejection, the Examiner argues,

Regarding Claims 4 & 5

Teachings of Mann '295 are disclosed in the claim 1 rejection above. Indexing table as interpreted above is a TLB for a cache memory.

Mann '295 does not teach that the block numbers in the indexing table are the same for the multiple different subsets of blocks.

Smith '1982 teaches that TLB having a hashing mechanism to map the virtual addresses (block numbers) to the real address (translated host code block) (Smith '1982: Pg.475 Col.2 Paragraphs 3-4). Hashing (done by taking an XOR or through randomized algorithm) depends on the number of bits selected, resulting in folding or overlapping (Smith '1982: Pg.488, Col.1 Paragraph 1; Pg.489, Col.1). The hashing scheme selected by the applicant is extremely simplified version, as only one middle bit is selected to index the translation indication index table.

It would have been obvious to one (e.g. a designer) of ordinary skill in the art at the time the invention was made to combine the teachings of Smith '1982 and apply them to Mann '295 to implement the indexing table as disclosed. The motivation would have been that Smith '1982 discloses the necessity for TLB like lookup when dealing with translated information when address space doesn't map directly (Smith '1982: Pg.510, Section 2.9, 2.17). Mann '295's design requires translation as one target code block may have one or more translated host code instructions. Hence Smith '1982 solves Mann '295's problem of mapping the target legacy instruction object to translated host object code block (Mann '295: Col.6 Lines 47-55).

10.2 In the Examiner's argument as quoted in Section 9.1 above, the Examiner relies upon the argument that an *Indexing table as interpreted above is a TLB for a cache memory.* Such interpretation is not believed accurate. No one skilled in the art would believe that a simple *indexing table* is interchangeable with a *TLB.* For example, the Examiner must agree that the *indexing table* of applicant could not be used to replace the *TLB* in the Smith '1982 publication. In a similar way, why would anyone adopt the complex structure of a TLB, with the large amounts of data implied, to the simple task of a small table in applicant's invention. In effect, all the attributes of a *TLB* are destroyed and the Smith '1982 *TLB* would no longer function as a *TLB*. In substance, the Examiner has relied upon the teaching in applicant's specification on how to

modify Smith '1982 to be a simple table. However, there is no teaching in Smith '1982 on how to make such modification nor any teaching as to any motivation for such modification. Those skilled in the art would readily recognize that a *TLB* and applicant's *indexing table* are not the same thing.

10.3 In the Examiner's argument as quoted in Section 9.1 above, the Examiner refers to (Mann '295: Col.6 Lines 47-55) as describing the problem solved by the *TLB* of Smith '1982. This argument is traversed to the extent that it can be understood.

10.4 The portion of Mann '295 in (Mann '295: Col.6 Lines 47-55) is as follows:

When alternate entry points into a block of code are detected, they are marked with an "E" or "Entry" Target code tag 72. As with the "F" or "First" Target code tag 72, the index/offset field 74 for the Target instruction 76 indexes/addresses 81 a block entry table 80, which in turn points to or addresses 83 the start of the corresponding Host code 88. The block entry tables 80 corresponding to a block of Host code 88 (or Target instructions 76) are chained together. In this FIG., a forward chain 85 is shown linking all of the block entry tables 80 in the block of code together. A link 84 is also shown linking the block entry tables 80 for secondary or alternate "Entry" points to the block entry table 80 for the main for "First" entry point into the block of code. Other methods of chaining or linking block entry tables 80 are also within the scope of this invention.

10.5 The portion of Mann '295 in (Mann '295: Col.6 Lines 47-55) is referring to part of the implementation associated with the TABLE T-1 of Col. 5 of Mann '295 and as stated in connection with FIG. 3, a forward chain 85 and link 84 are employed to link entry points. Applicant cannot understand how the chained structure of FIG. 3 in Mann '295 as described in Mann '295: Col.6 Lines 47-55 presents any problem that is solved by a TLB of Smith '1982. Therefore, the argument by the Examiner that there is some motivation to combine the Mann '295 and Smith '1982 references is not believed well founded.

10.6 Furthermore, even if the Mann '295 and Smith '1982 references are combined as suggested by the Examiner, the combination does not suggest applicant's invention. Mann '295 discusses the problem of self-modifying code at Col. 7 Lines 4-48 (and similarly at Col. 9 Lines 5-20) and the method employed by Mann '295 is merely to abort and return to the interpreter so that re-translation is required in every case. Such operation is in marked contrast to the present invention where re-translation is avoided when unnecessary leading to a great savings in execution time. Even if the Examiner used the combination of Mann '295 and Smith '1982, such combination does not describe the avoidance of re-translation if instruction data has not been changed.

10.7 Claim 1 and the other independent claims after the amendments of this *Response A* now all include the limitations of the Claim 4 type together with other clarifying amendments. Accordingly, upon reconsideration, of the claims as amended, it is believed that Examiner will now find that all the claims are allowable.

11 The Examiner rejected Claims 6 & 20 under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No. 6,516,295 issued to George A. Mann et al (Mann '295 hereafter) in view of U.S. Patent No. 5,560,013 issued to Casper A. Scalzi et al (Scalzi '013 hereafter).

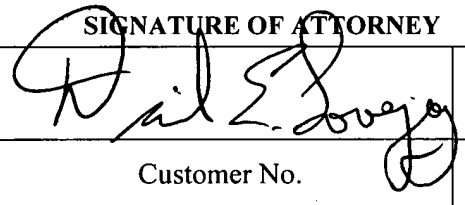
11.1 This rejection is traversed for all of the reasons discussed above in connection with independent Claim 1 from which Claims 6 and 10 depend. Claims 6 and 10 are believed allowable for the same reasons that Claim 1 is allowable.

12 The Examiner rejected Claims 14 & 28 under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No. 6,516,295 issued to George A. Mann et al (Mann '295 hereafter) in view of ACM Article "Cache Memories" by Alan Jay Smith (Smith '1982 hereafter), further in view of U.S. Patent No. 5,560,013 issued to Casper A. Scalzi et al (Scalzi '013 hereafter).

12.1 This rejection is traversed for all of the reasons discussed above in Sections 9 and 10 applicable to all independent claims. Claims 14 & 28 are believed allowable for the same reasons discussed above in Sections 9 and 10.

13 The Examiner rejected Claims 8 & 22 under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No. 6,516,295 issued to George A. Mann et al (Mann '295 hereafter) in view of applicant's own admission in the specification.

13.1 This rejection is traversed for all of the reasons discussed above in Sections 9 and 10 applicable to all claims. Claims 8 & 22 are believed allowable for the same reasons discussed above in Sections 9 and 10.

SIGNATURE OF ATTORNEY		
David E. Lovejoy (Reg. No. 22,748)		**Signature Date:** 30 December 2005
Address	Customer No.	Communication
102 Reed Ranch Rd. Tiburon, CA 94920-2025	21603	Tel: (415) 435-8203 Fax: (415) 435-8857 e-mail: david.lovejoy@sbcglobal.net

Amendments to the Drawings

The attached sheet of drawings includes changes to FIG. 2. This sheet, which includes FIG. 1and FIG. 2, replaces the original sheet including FIG. 1and FIG. 2. In FIG. 2, previously Numbered elements 22 and 23 have been reversed to conform to the specification.

Attachments:

Replacement Sheet

Annotated Replacement Sheet

INTERLINED CLAIMS AFTER *RESPONSE A*

1. (First Amended per A) A computer implemented method for emulating execution of legacy instructions, where said legacy instructions have instruction addresses, comprising:

accessing blocks of said legacy instructions, said blocks having block addresses,

storing translations, into a translation store ~~translation information,~~ for each of the legacy instructions,

storing translation indications, for indicating translated blocks, into an indexing table at block numbers determined by said block addresses,

~~for each particular~~said storing translation indications using a subset of block address digits whereby block numbers in said table are the same for multiple different blocks,

executing said translated instructions to emulate said legacy ~~instruction~~instructions,

where for each of the legacy instructions of a translated block having a ~~particular~~ block number in said table,

said storing translations step includes translating the ~~particular~~ legacy instructions into one or more translated instructions for emulating the ~~particular~~legacy instructions, and

if the legacy instruction~~,~~

~~if the particular~~ is not a store instruction, going to said step of executing said translated instructions,

if the legacy instruction is a store instruction, where the store instruction stores to a particular block with a particular block number in said table, checking the indications in said table for ~~said~~the particular block number ~~to determine~~and,

if the ~~if instruction data has been stored for~~ndications indicate that said particular block ~~number,~~

~~if instruction data has been stored for~~has not been translated, going to said step of executing said translated instructions,

if the indications indicate that said particular block ~~number~~has been translated, checking said translation

store to determine if legacy instruction data has been modified and if modified, repeating the step of translating the legacy instructions and going to said step of executing said translated instructions; and otherwise, if legacy instruction data has not been ~~stored for said particular block number, bypassing said checking.~~ modified, going to said step of executing said translated instructions.

INTERLINED CLAIMS AFTER *RESPONSE A*

2. (Original) The method of Claim 1 wherein said step of storing translation indications stores indications for only a subset of all the translated blocks.

3. (Original) The method of Claim 2 wherein said subset of all the translated blocks is stored in a cache.

4. (Cancelled).

5. (First Amended per A) The method of Claim ~~4~~ 1 wherein said block address digits are included in a three digit hexadecimal address field and said subset of block address digits is the center digit.

6. (Original) The method of Claim 1 wherein said legacy instructions are for a legacy system having a S/390 architecture.

7. (Original) The method of Claim 1 wherein said legacy instructions are object code instructions compiled/assembled for a legacy architecture.

8. (Original) The method of Claim 1 wherein said legacy instructions include store instructions for modifying instruction code.

9. (Original) The method of Claim 1 wherein said translation indications include a state field for each block number indicating whether the block represented by said block number has been modified.

INTERLINED CLAIMS AFTER *RESPONSE A*

10. (First Amended per A) The method of Claim 1 wherein,

~~said step of storing translation indications stores indications for only a subset of all the translated blocks and uses a subset of block address digits whereby block numbers in said table are the same for multiple different blocks,~~

said subset of all the translated blocks is stored in a cache,

said translation indications include a state field storing a count for each block number indicating whether the block represented by said block number has been modified,

said count in a state field is incremented each time a block represented by said block number has been modified in said cache,

said count in a state field is decremented each time a block represented by said block number has been removed from said cache,

said ~~bypassing said~~step of checking ~~step~~said translation store occurs only when said count is zero.

11. (First Amended per A) A computer implemented method for dynamic emulation of object code legacy instructions, where the legacy instructions have instruction addresses determined by compilation/assembly of source code and where the legacy instructions include self-modifying store instructions for modifying instruction code, comprising:

accessing blocks of said legacy instructions, said blocks having block addresses,

storing translations, into a translation store ~~translation information~~, for each of the legacy instructions,

storing translation indications, for only a subset of all the translated blocks, into an indexing table at block numbers determined by said block addresses, said storing translation indications,

using a subset of block address digits whereby block numbers in said table are the same for multiple different blocks,

including a state field storing a count for each block number indicating whether the block represented by said block number has been modified by self-modifying store instructions,

executing said translated instructions to emulate said legacy instructions,

where for each ~~particular~~of the legacy instructions of said subset of all the translated blocks having a ~~particular~~ block number in said table,

said storing translations step includes translating the ~~particular~~ legacy instruction into one or more translated instructions for emulating the ~~particular~~ legacy instruction;

and storing said translated instructions in a cache,

if the ~~particular~~legacy instruction is not a store instruction, going to said step of executing said translated instructions,

if the legacy instruction is a store instruction, where the store instruction stores to a particular block with a particular block number in said

table, checking the indications in said table for said particular block number ~~to determine~~and,

if the ~~if instruction data has been stored for~~ndications indicate that said particular block ~~number,~~

~~if instruction data has been stored for~~has not been translated, going to said step of executing said translated instructions,

if the indications indicate that said particular block ~~number~~has been translated, checking said translation store to determine if legacy instruction data has been modified and if modified, repeating the step of translating the legacy instructions and going to said step of executing said translated instructions; and otherwise, if instruction data has not been ~~stored for said particular block number, bypassing said checking.~~ modified going to said step of executing said translated instructions.

12. (First Amended per A) The method of Claim 11 wherein said count in a state field is incremented each time a block represented by said block number has been modified in said cache, said count in a state field is decremented each time a block represented by said block number has been removed from said cache, said ~~bypassing said~~step of checking ~~step~~said translation store occurs only when said count is zero.

13. (Original) The method of Claim 11 wherein said legacy code is compiled/assembled for a native architecture and executes as a guest on a host architecture.

14. (Original) The method of Claim 13 wherein the native architecture employs CISC instructions and the host architecture employs RISC instructions.

INTERLINED CLAIMS AFTER *RESPONSE A*

15. (First Amended per A) A computer system for emulating execution of legacy instructions, where said legacy instructions have instruction addresses, comprising:

a group access unit for accessing blocks of said legacy instructions, said blocks having block addresses,

a translator for translating the legacy instructions to form translated instructions,

a translation store for storing ~~translation information for each of the~~the translated instructions,

an execution unit for executing said translated instructions to emulate said legacy instructions,

an index table for storing translation indications~~,~~ for indicating translated blocks at block numbers determined by said block addresses, said index table storing translation indications using a subset of block address digits whereby block numbers in said table are the same for multiple different blocks,

where for each ~~particular~~of the legacy instruction of a translated block having a ~~particular~~ block number in said table,

~~to translate the particular legacy instruction into~~ said translation store includes one or more translated instructions for emulating the ~~particular~~legacy instruction, and,

if the legacy instruction~~,~~

~~if the particular~~ is not a store instruction, the computer system goes to the execution unit for executing said translated instructions,

if the legacy instruction is a store instruction, ~~to~~where the store instruction stores to a particular block with a particular block number in said table, the computer system checks the indications in said table for said particular block number ~~to determine~~and,

if the ~~if instruction data has been stored for~~ndications indicate that said particular block ~~number,~~

~~if instruction data has been stored for~~has not been translated, the computer system goes to the execution unit for executing said translated instructions,

if the indications indicate that said particular block ~~number, to check~~has not been translated, said translation store is checked to determine if instruction data has been modified; and, if modified, the translator repeats translating the legacy instructions and the computer system goes to the execution unit for executing said translated instructions, and otherwise, if instruction data has not been ~~stored for said particular block number, to bypass said checking.~~ modified, the computer system goes to the execution unit for executing said translated instructions.

16. (Original) The system of Claim 15 wherein said index table stores indications for only a subset of all the translated blocks.

17. (Original) The system of Claim 16 including a cache and wherein said subset of all the translated blocks is stored in said cache.

18. (Cancelled).

19. (First Amended per A) The system of Claim 1~~8~~5 wherein said block address digits are included in a three digit hexadecimal address field and said subset of block address digits is the center digit.

20. (Original) The system of Claim 15 wherein said legacy instructions are for a legacy system having a S/390 architecture.

INTERLINED CLAIMS AFTER *RESPONSE A*

21. (Original) The system of Claim 15 wherein said legacy instructions are object code instructions compiled/assembled for a legacy architecture.

22. (Original) The system of Claim 15 wherein said legacy instructions include store instructions for modifying instruction code.

23. (Original) The system of Claim 15 wherein said index table includes a state field for each block number indicating whether the block represented by said block number has been modified.

24. (First Amended per A) The system of Claim 15 wherein,

~~said index table stores indications for only a subset of all the translated blocks and uses a subset of block address digits whereby block numbers in said table are the same for multiple different blocks,~~

~~said subset of all the translated blocks,~~

said system includes a cache for storing said subset of all the translated blocks,

said index table includes a state field storing a count for each block number indicating whether the block represented by said block number has been modified,

said count in a state field is incremented each time a block represented by said block number has been modified in said cache,

said count in a state field is decremented each time a block represented by said block number has been removed from said cache,

said ~~bypassing of said checking occurs~~translation store is not checked only when said count is zero.

INTERLINED CLAIMS AFTER *RESPONSE A*

25. (First Amended per A) A computer system for dynamic emulation of object code legacy instructions, where the legacy instructions have instruction addresses determined by compilation/assembly of source code and where the legacy instructions include self-modifying store instructions for modifying instruction code, comprising:

a group access unit for accessing blocks of said legacy instructions, said blocks having block addresses,

~~storing into~~ a translation store for storing translation information for each of the legacy instructions,

an index table for storing translation indications, for only a subset of all the translated blocks ~~—————~~at block numbers determined by said block addresses, said index table storing translation indications~~;~~

~~————————~~ using a subset of block address digits whereby block numbers in said table are the same for multiple different blocks~~;~~

~~————————~~ and including a state field storing a count for each block number indicating whether the block represented by said block number has been modified by self-modifying store instructions,

a cache for storing translated instructions,

an execution unit for executing said translated instructions to emulate said legacy instructions,

a legacy code translator operating, for each ~~particular~~of the legacy instructions of said subset of all the translated blocks having a ~~particular~~ block number in said table,

~~————————~~ to translate the ~~particular~~ legacy instructions into one or more translated instructions for emulating the ~~particular~~ legacy instruction~~;~~

~~————————~~ ~~To~~ and to store said translated instructions in athe cache and,

if the ~~particular~~legacy instruction is not a store instruction, the computer system goes to said execution unit for executing said translated instructions.

if the legacy instruction is a store instruction, ~~to~~where the store instruction stores to a particular block with a particular block number in said table, the computer system checks the indications in said table for said particular block number ~~to determine~~and,

if the ~~if instruction data has been stored for~~ndications indicate that said particular block ~~number,~~

~~if instruction data has been stored for~~has not been translated, the computer system goes to said execution unit for executing said translated instructions,

if the indications indicate that said particular block ~~number, checking said translation store~~has been translated, the computer system checks to determine if instruction data has been modified and if modified, the computer system goes to said translator to translate the legacy instructions into one or more translated instructions and the computer system goes to said execution unit for executing said translated instructions; and otherwise, if instruction data has not been ~~stored for said particular block number, to bypass said checking.~~ modified, the computer system goes to said execution unit for executing said translated instructions.

26. (First Amended per A) The system of Claim 25 wherein said count in a state field is incremented each time a block represented by said block number has been modified in said cache, said count in a state field is decremented each time a block represented by said block number has been removed from said cache, said bypassing said ~~checking step occurs~~repeating occurring only when said count is zero.

27. (Original) The system of Claim 25 wherein said legacy code is compiled/assembled for a native architecture and executes as a guest on a host architecture.

28. (Original) The system of Claim 27 wherein the native architecture employs CISC instructions and the host architecture employs RISC instructions.

FIG. 1

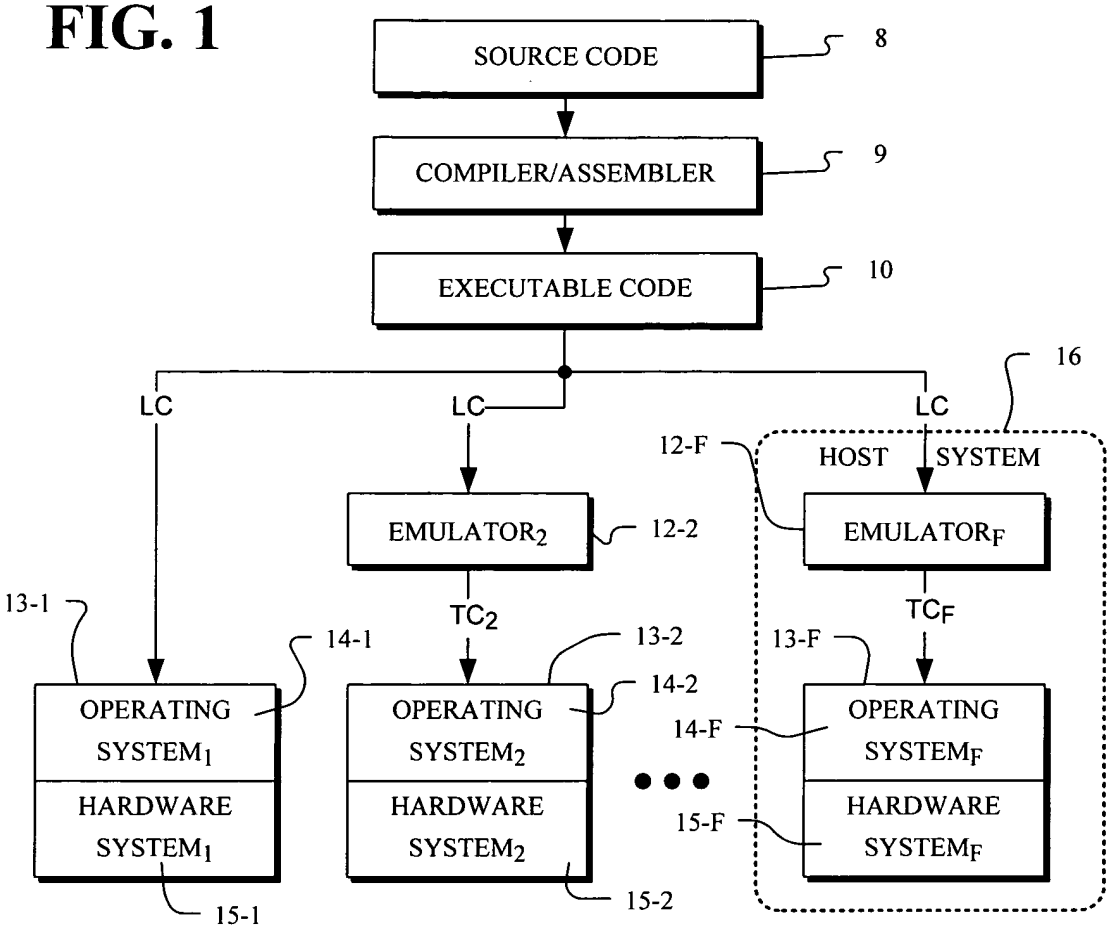


FIG. 2